Computational Perspectives on Logical Foundations

Dominik Kirst

Application talk Inria Université Côte d'Azur, Stamp team

Past Activities

CV Overview

CV Overview

Saarland University (2011-2015)

Bachelor's thesis with Gert Smolka

CV Overview

Saarland University (2011-2015)

Bachelor's thesis with Gert Smolka

Oxford University (2015-2016)

Master's thesis with Luke Ong

CV Overview

Saarland University (2011-2015)

Bachelor's thesis with Gert Smolka

Oxford University (2015-2016)

Master's thesis with Luke Ong

Saarland University (2016-2023)

PhD thesis with Gert Smolka

CV Overview

Saarland University (2011-2015)

Bachelor's thesis with Gert Smolka

Oxford University (2015-2016)

Master's thesis with Luke Ong

Saarland University (2016-2023)

PhD thesis with Gert Smolka

Ben-Gurion University (2023-2024)

Minerva Fellow with Liron Cohen

CV Overview

Saarland University (2011-2015)

Bachelor's thesis with Gert Smolka

Oxford University (2015-2016)

Master's thesis with Luke Ong

Saarland University (2016-2023)

PhD thesis with Gert Smolka

Ben-Gurion University (2023-2024)

Minerva Fellow with Liron Cohen

Inria/IRIF, Picube (since 2024)

Marie Curie Fellow with Hugo Herbelin

Three Overlapping Research Fields

Coq Proof Assistant

Dependent Type Theory

Constructive Mathematics

Field 1: Coq (Rocq) Proof Assistant

- Mechanised mathematics: four colour theorem, odd-order theorem, value of BB(5), ...
- Verification of realistic software: CompCert, CertiCoq, Iris, ...
- Applicable in teaching logic, arithmetics, calculus, ...

Contribution 1: Coq Library for Mechanised First-Order Logic

mark-koch Fix Proofmode MinZF demo ✓ 2722b67 4 days ago **History**

..		
Arithmetics	Rename Deduction -> ND to prepare for Sequent	2 months ago
Completeness2	Start using Asimpl in places	5 days ago
Deduction	Finish Kripke Completeness, work on atom substitution	10 days ago
Incompleteness	Tarski Constructions ported	2 months ago
Proofmode	Fix Proofmode MinZF demo	4 days ago
Reification	Tarski Constructions ported	2 months ago
Semantics	Add validity facts, remove "not _strong" preservation	10 days ago
Sets	Rename Deduction -> ND to prepare for Sequent	2 months ago
Syntax	Start using Asimpl in places	5 days ago
Undecidability	Start using Asimpl in places	5 days ago
Utils	Port FOLP reduction, refactor PCP decidabilities	2 months ago
FragmentSyntax.v	Start using Asimpl in places	5 days ago
FullSyntax.v	Start using Asimpl in places	5 days ago

Contribution 1: Coq Library for Mechanised First-Order Logic

mark-koch Fix Proofmode MinZF demo ✓ 2722b67 4 days ago **History**

..		
Arithmetics	Rename Deduction -> ND to prepare for Sequent	2 months ago
Completeness2	Start using Asimpl in places	5 days ago
Deduction	Finish Kripke Completeness, work on atom substitution	10 days ago
Incompleteness	Tarski Constructions ported	2 months ago
Proofmode	Fix Proofmode MinZF demo	4 days ago
Reification	Tarski Constructions ported	2 months ago
Semantics	Add validity facts, remove "not _strong" preservation	10 days ago
Sets	Rename Deduction -> ND to prepare for Sequent	2 months ago
Syntax	Start using Asimpl in places	5 days ago
Undecidability	Start using Asimpl in places	5 days ago
Utils	Port FOLP reduction, refactor PCP decidabilities	2 months ago
FragmentSyntax.v	Start using Asimpl in places	5 days ago
FullSyntax.v	Start using Asimpl in places	5 days ago

- Contents: CPP'19, LFCS'20, IJCAR'20, ITP'21, ITP'22, FSCD'22, CSL'23, CPP'24, CSL'25, LICS'25

Contribution 1: Coq Library for Mechanised First-Order Logic

mark-koch Fix Proofmode MinZF demo ✓ 2722b67 4 days ago **History**

..		
Arithmetics	Rename Deduction -> ND to prepare for Sequent	2 months ago
Completeness2	Start using Asimpl in places	5 days ago
Deduction	Finish Kripke Completeness, work on atom substitution	10 days ago
Incompleteness	Tarski Constructions ported	2 months ago
Proofmode	Fix Proofmode MinZF demo	4 days ago
Reification	Tarski Constructions ported	2 months ago
Semantics	Add validity facts, remove "not _strong" preservation	10 days ago
Sets	Rename Deduction -> ND to prepare for Sequent	2 months ago
Syntax	Start using Asimpl in places	5 days ago
Undecidability	Start using Asimpl in places	5 days ago
Utils	Port FOLP reduction, refactor PCP decidabilities	2 months ago
FragmentSyntax.v	Start using Asimpl in places	5 days ago
FullSyntax.v	Start using Asimpl in places	5 days ago

- Contents: CPP'19, LFCS'20, IJCAR'20, ITP'21, ITP'22, FSCD'22, CSL'23, CPP'24, CSL'25, LICS'25
- Tool support for common implementation challenges and external usability

Field 2: Dependent Type Theory

- Richly typed functional programming language
- Interpretation as logical system base for proof assistants
- Natural connection of computation and logic

Contribution 2: Synthetic Approach to Computability Theory

Definition (Traditional)

A set $X \subseteq \mathbb{N}$ is decidable if there is a function $f : X \to \mathbb{B}$ such that

$$x \in X \;\leftrightarrow\; f(x) = \mathsf{tt}$$

and there is a total Turing machine M computing f: $\forall x.\, f(x) = \mathsf{tt} \leftrightarrow M[x] \downarrow 1$.

Contribution 2: Synthetic Approach to Computability Theory

Definition (Synthetic)

A set $X \subseteq \mathbb{N}$ is decidable if there is a function $f : X \to \mathbb{B}$ such that

$$x \in X \;\leftrightarrow\; f(x) = \mathsf{tt}$$

~~and there is a total Turing machine M computing f: $\forall x.\, f(x) = \mathsf{tt} \leftrightarrow M[x] \downarrow 1$.~~

Contribution 2: Synthetic Approach to Computability Theory

Definition (Synthetic)

A set $X \subseteq \mathbb{N}$ is decidable if there is a function $f : X \to \mathbb{B}$ such that

$$x \in X \;\leftrightarrow\; f(x) = \mathsf{tt}$$

~~and there is a total Turing machine M computing f: $\forall x.\, f(x) = \mathsf{tt} \leftrightarrow M[x] \downarrow 1$.~~

Simplification of computability theory without Turing machines:

Contribution 2: Synthetic Approach to Computability Theory

Definition (Synthetic)

A set $X \subseteq \mathbb{N}$ is decidable if there is a function $f : X \to \mathbb{B}$ such that

$$x \in X \;\leftrightarrow\; f(x) = \mathsf{tt}$$

~~and there is a total Turing machine M computing f: $\forall x.\, f(x) = \mathsf{tt} \leftrightarrow M[x] \downarrow 1$.~~

Simplification of computability theory without Turing machines:

- **Undecidability**: validity (CPP'19), finite satisfiability (IJCAR'20), dyadic syntax (ITP'22)

Contribution 2: Synthetic Approach to Computability Theory

Definition (Synthetic)

A set $X \subseteq \mathbb{N}$ is decidable if there is a function $f : X \to \mathbb{B}$ such that

$$x \in X \;\leftrightarrow\; f(x) = \mathsf{tt}$$

~~and there is a total Turing machine M computing f: $\forall x.\, f(x) = \mathsf{tt} \leftrightarrow M[x] \downarrow 1$.~~

Simplification of computability theory without Turing machines:

- **Undecidability**: validity (CPP'19), finite satisfiability (IJCAR'20), dyadic syntax (ITP'22)
- **Incompleteness**: PA/ZF (ITP'21), Tennenbaum's thm. (FSCD'22), abstraction (CSL'23)

Contribution 2: Synthetic Approach to Computability Theory

Definition (Synthetic)

A set $X \subseteq \mathbb{N}$ is decidable if there is a function $f : X \to \mathbb{B}$ such that

$$x \in X \;\leftrightarrow\; f(x) = \mathsf{tt}$$

~~and there is a total Turing machine M computing f: $\forall x.\, f(x) = \mathsf{tt} \leftrightarrow M[x] \downarrow 1$.~~

Simplification of computability theory without Turing machines:

- **Undecidability**: validity (CPP'19), finite satisfiability (IJCAR'20), dyadic syntax (ITP'22)
- **Incompleteness**: PA/ZF (ITP'21), Tennenbaum's thm. (FSCD'22), abstraction (CSL'23)
- **Oracles**: definitions (APLAS'23), Post's thm. (CSL'24), Post's problem (TYPES'24)

Field 3: Constructive Mathematics

- Alternative foundation of mathematics based on more modest assumptions
  - Unveils hidden building blocks of mathematical constructions
    - Generalises connection of computation and logic

Contribution 3: Constructive Analysis of the Completeness Theorem

Theorem

If a first-order formula φ is valid in all models ($\vDash \varphi$) then it is provable ($\vdash$).

Contribution 3: Constructive Analysis of the Completeness Theorem

Theorem

If a first-order formula φ is valid in all models ($\vDash \varphi$) then it is provable ($\vdash$).

- Completeness Theorem equivalent to Boolean Prime Ideal Theorem (Henkin, 1954)
- Completeness Theorem equivalent to Markov's Principle (Kreisel, 1962)
- Completeness Theorem equivalent to Weak König's Lemma (Simpson, 2009)
- Completeness Theorem equivalent to Weak Fan Theorem (Krivtsov, 2015)

Contribution 3: Constructive Analysis of the Completeness Theorem

Theorem

If a first-order formula φ is valid in all models ($\vDash \varphi$) then it is provable ($\vdash$).

- Completeness Theorem equivalent to Boolean Prime Ideal Theorem (Henkin, 1954)
- Completeness Theorem equivalent to Markov's Principle (Kreisel, 1962)
- Completeness Theorem equivalent to Weak König's Lemma (Simpson, 2009)
- Completeness Theorem equivalent to Weak Fan Theorem (Krivtsov, 2015)
- Completeness Theorem holds without any assumptions (Krivine, 1996)

Contribution 3: Constructive Analysis of the Completeness Theorem

Theorem

If a first-order formula φ is valid in all models ($\vDash \varphi$) then it is provable ($\vdash$).

- Completeness Theorem equivalent to Boolean Prime Ideal Theorem (Henkin, 1954)
- Completeness Theorem equivalent to Markov's Principle (Kreisel, 1962)
- Completeness Theorem equivalent to Weak König's Lemma (Simpson, 2009)
- Completeness Theorem equivalent to Weak Fan Theorem (Krivtsov, 2015)
- Completeness Theorem holds without any assumptions (Krivine, 1996)

Ongoing systematic investigation inspired by Herbelin and Ilik (2016):

Contribution 3: Constructive Analysis of the Completeness Theorem

Theorem

If a first-order formula φ is valid in all models ($\vDash \varphi$) then it is provable ($\vdash$).

- Completeness Theorem equivalent to Boolean Prime Ideal Theorem (Henkin, 1954)
- Completeness Theorem equivalent to Markov's Principle (Kreisel, 1962)
- Completeness Theorem equivalent to Weak König's Lemma (Simpson, 2009)
- Completeness Theorem equivalent to Weak Fan Theorem (Krivtsov, 2015)
- Completeness Theorem holds without any assumptions (Krivine, 1996)

Ongoing systematic investigation inspired by Herbelin and Ilik (2016):

- Consolidation (LFCS'20) $\Rightarrow$ new observations (LFCS'22) $\Rightarrow$ systematisation (TYPES'23)

Contribution 3: Constructive Analysis of the Completeness Theorem

Theorem

If a first-order formula φ is valid in all models ($\vDash \varphi$) then it is provable ($\vdash$).

- Completeness Theorem equivalent to Boolean Prime Ideal Theorem (Henkin, 1954)
- Completeness Theorem equivalent to Markov's Principle (Kreisel, 1962)
- Completeness Theorem equivalent to Weak König's Lemma (Simpson, 2009)
- Completeness Theorem equivalent to Weak Fan Theorem (Krivtsov, 2015)
- Completeness Theorem holds without any assumptions (Krivine, 1996)

Ongoing systematic investigation inspired by Herbelin and Ilik (2016):

- Consolidation (LFCS'20) $\Rightarrow$ new observations (LFCS'22) $\Rightarrow$ systematisation (TYPES'23)
- Generalisation (CPP'24) and semantic clarification (LICS'24)

Publications Overview

- Conference papers (APLAS, CPP, CSL, FSCD, IJCAR, ITP, LFCS, LICS, WoLLIC, 21 in total)
- Journal articles (JAR, LMCS, MSCS, Journal of Logic and Computation, 7 in total)
- Workshop abstracts (AAL, ALC, CCC, CoqPL, CoqWS, HoTT/UF, TYPES, 17 in total)
- International research network with 20 co-authors in total
- All projects are accompanied by Coq developments

Publications Overview

- Conference papers (APLAS, CPP, CSL, FSCD, IJCAR, ITP, LFCS, LICS, WoLLIC, 21 in total)
- Journal articles (JAR, LMCS, MSCS, Journal of Logic and Computation, 7 in total)
- Workshop abstracts (AAL, ALC, CCC, CoqPL, CoqWS, HoTT/UF, TYPES, 17 in total)
- International research network with 20 co-authors in total
- All projects are accompanied by Coq developments

Visibility led to invited seminar talks, invited lectures at summer schools, service on PCs, features on a podcast

Software Development

Software Development

Coq Library for First-Order Logic (Leader):

- Meta-theory of FOL syntax and semantics including tool support
- 11 contributors and more than 25k loc

Software Development

Coq Library for First-Order Logic (Leader):

- Meta-theory of FOL syntax and semantics including tool support
- 11 contributors and more than 25k loc

Coq Library of Undecidability Proofs (Contributor):

- Mechanised reductions establishing undecidability results in several domains
- 16 contributors and more than 100k loc

Software Development

Coq Library for First-Order Logic (Leader):

- Meta-theory of FOL syntax and semantics including tool support
- 11 contributors and more than 25k loc

Coq Library of Undecidability Proofs (Contributor):

- Mechanised reductions establishing undecidability results in several domains
- 16 contributors and more than 100k loc

Coq Library for Synthetic Computability (Contributor):

- Synthetic development of computability theory
- 7 contributors and more than 20k loc

Teaching and Supervision Experience

Teaching and Supervision Experience

At Saarland University:

- Student TA in several courses already during undergraduate studies
- Main organiser of basic and advanced lectures (up to 600 students)
- Conception of own seminars for specialised topics
- 11 supervised bachelor's students
- 2 supervised master's students
- 3 supervised engineering internships

Teaching and Supervision Experience

At Saarland University:

- Student TA in several courses already during undergraduate studies
- Main organiser of basic and advanced lectures (up to 600 students)
- Conception of own seminars for specialised topics
- 11 supervised bachelor's students
- 2 supervised master's students
- 3 supervised engineering internships

At Inria Paris:

- 1 supervised M2 internship
- 1 ongoing M2 internship
- Outreach via SIGPLAN-M mentoring program

Research Project:
Unified Reverse Mathematics

State of the Art

State of the Art

Classical reverse mathematics

Understand the logical strength of mathematical theorems

State of the Art

Classical reverse mathematics

Axioms $\Rightarrow$ Theorems

Understand the logical strength of mathematical theorems

State of the Art

Classical reverse mathematics

Axioms $\iff$ Theorems

Understand the logical strength of mathematical theorems

State of the Art

Classical reverse mathematics

Axioms $\Longleftrightarrow$ Theorems

Understand the logical strength of mathematical theorems

Constructive reverse mathematics

Understand the computational strength of mathematical theorems

State of the Art

Classical reverse mathematics

Axioms $\Longleftrightarrow$ Theorems

Understand the logical strength of mathematical theorems

Constructive reverse mathematics

Computations $\Rightarrow$ Theorems

Understand the computational strength of mathematical theorems

State of the Art

Classical reverse mathematics

Axioms $\iff$ Theorems

Understand the logical strength of mathematical theorems

Constructive reverse mathematics

Computations $\iff$ Theorems

Understand the computational strength of mathematical theorems

State of the Art

Classical reverse mathematics

Axioms $\iff$ Theorems

Understand the logical strength of mathematical theorems

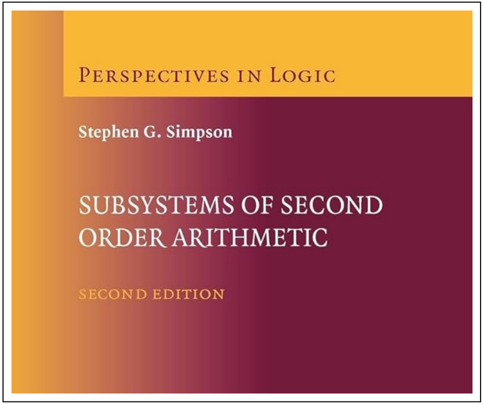
$$\frac{\text{In principle shared motivation but}}{\text{incompatible base systems, different evaluation dimensions, sociological misunderstandings}}$$

Constructive reverse mathematics

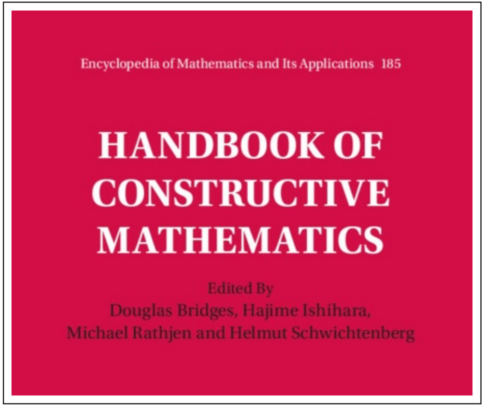
Computations $\iff$ Theorems

Understand the computational strength of mathematical theorems

State of the Art

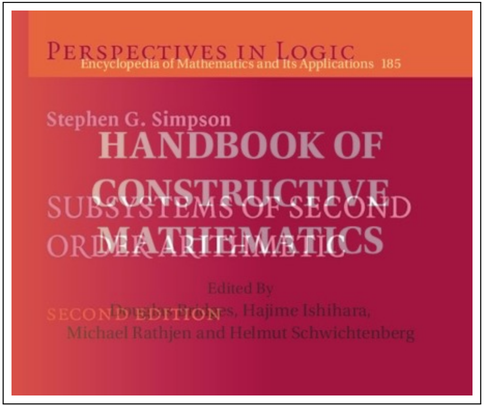


Classical Reverse Mathematics



Constructive Reverse Mathematics

Vision



Unified Reverse Mathematics

Vision

Unified reverse mathematics

Axioms/Computations $\Leftrightarrow$ Theorems

Vision

Unified reverse mathematics

Axioms/Computations $\Leftrightarrow$ Theorems

The previous challenges can be overcome by a combination of:

Vision

Unified reverse mathematics

Axioms/Computations $\Leftrightarrow$ Theorems

The previous challenges can be overcome by a combination of:

- **Constructive type theories like CIC**
  Embodying agnostic logical foundations suitable for both traditions,
  Integrating both logical and computational perspectives

Vision

Unified reverse mathematics

Axioms/Computations $\Leftrightarrow$ Theorems

The previous challenges can be overcome by a combination of:

- **Constructive type theories like CIC**
  Embodying agnostic logical foundations suitable for both traditions,
  Integrating both logical and computational perspectives

- **Proof assistants like Coq implementing CIC**
  Providing support in the exploration and verification of new results,
  Implementing ideal tools for reverse mathematics

Objectives

Objectives

1. Unify existing reverse mathematical results

Objectives

1. Unify existing reverse mathematical results
2. Transfer results from one setting to the other

Objectives

1. Unify existing reverse mathematical results
2. Transfer results from one setting to the other
3. Develop a proof assistant (variant) for the respective base systems

Objectives

1. Unify existing reverse mathematical results
2. Transfer results from one setting to the other
3. Develop a proof assistant (variant) for the respective base systems
4. Implement a comprehensive formal library in it

Objectives

1. Unify existing reverse mathematical results
2. Transfer results from one setting to the other
3. Develop a proof assistant (variant) for the respective base systems
4. Implement a comprehensive formal library in it
5. Clarify the logical foundations of the respective base systems

Impact

Impact

Theoretical impact:

- Long-sought reconciliation of classical and constructive reverse mathematics
- Deeper computational understanding of logical foundations
- Axiom minimisation generalises theorems to novel applications

Impact

Theoretical impact:

- Long-sought reconciliation of classical and constructive reverse mathematics
- Deeper computational understanding of logical foundations
- Axiom minimisation generalises theorems to novel applications

Technological impact:

- Contribution to vision pursued by formal libraries like MathComp
- Extensions of meta-programming tools like Elpi and MetaCoq
- Axiom minimisation improves interoperability, efficiency, automation

Impact

Theoretical impact:

- Long-sought reconciliation of classical and constructive reverse mathematics
- Deeper computational understanding of logical foundations
- Axiom minimisation generalises theorems to novel applications

Technological impact:

- Contribution to vision pursued by formal libraries like MathComp
- Extensions of meta-programming tools like Elpi and MetaCoq
- Axiom minimisation improves interoperability, efficiency, automation

Dissemination:

- Theory-oriented venues like CSL, FSCD, LICS, WoLLIC
- Mechanisation-focused venues like CPP, IJCAR, ITP, JAR

Integration at Inria

Integration with Stamp

Verification of algorithms and mathematical results:
Proof assistants, formal libraries, meta-programming, . . .

Integration with Stamp

Verification of algorithms and mathematical results:
Proof assistants, formal libraries, meta-programming, . . .

- Yves Bertot: one of the main contributors to Coq, coordinator of LiberAbaci
- Enrico Tassi: formal libraries like MathComp and meta-programming in Elpi
- Laurent Théry: mechanised mathematics in MathComp Analysis

Integration with Stamp

Verification of algorithms and mathematical results:
Proof assistants, formal libraries, meta-programming, ...

- Yves Bertot: one of the main contributors to Coq, coordinator of LiberAbaci
- Enrico Tassi: formal libraries like MathComp and meta-programming in Elpi
- Laurent Théry: mechanised mathematics in MathComp Analysis
- Dominik Kirst: reverse mathematics, models of type theory, mechanised comp. physics

External Collaborations

- Aix-Marseille University *(France)*: Étienne Miquey
- Australian National University *(Australia)*: Ian Shillito
- Ben-Gurion University *(Israel)*: Liron Cohen, Ariel Grunfeld
- Birmingham University *(United Kingdom)*: Vincent Rahli, Bruno da Rocha Paiva
- ETH Zürich *(Switzerland)*: Johannes Hostert
- Gothenburg University *(Sweden)*: Dominik Wehr
- Inria Paris *(France)*: Hugo Herbelin, Yannick Forster
- Lorraine University *(France)*: Dominique Larchey-Wendling
- Max-Planck Institute SWS *(Germany)*: Niklas Mück, Benjamin Peters
- Oxford University *(United Kingdom)*: Janis Bailitis, Christian Hagemeier, Mark Koch
- Radboud University *(Netherlands)*: Marc Hermes
- Saarland University *(Germany)*: Gert Smolka, Haoyi Zeng
- TU Dortmund *(Germany)*: Andrej Dudenhefner

Summary

Profile: Computational Perspectives on Logical Foundations

Coq Proof Assistant

Dependent Type Theory

Constructive Mathematics

Profile: Computational Perspectives on Logical Foundations

Coq Proof Assistant

Dependent Type Theory

Constructive Mathematics

$\Downarrow$

Research project: Unified Reverse Mathematics

Profile: Computational Perspectives on Logical Foundations

Coq Proof Assistant

Dependent Type Theory

Constructive Mathematics

⇓

Research project: Unified Reverse Mathematics

⇑

Theoretical research (28 full papers, 20 international co-authors)

Software development (Coq library for first-order logic, tool support)

Bibliography I

Cohen, L., Forster, Y., Kirst, D., da Rocha Paiva, B., and Rahli, V. (2024). Separating Markov's principles. In *39th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'24), July 8–11, 2024, Tallinn, Estonia*.

Forster, Y., Kirst, D., and Wehr, D. (2021). Completeness theorems for first-order logic analysed in constructive type theory: Extended version. *Journal of Logic and Computation*, 31(1):112–151.

Hagemeier, C. and Kirst, D. (2022). Constructive and mechanised meta-theory of IEL and similar modal logics. *Journal of Logic and Computation*, 32(8):1585–1610.

Henkin, L. (1954). Metamathematical theorems equivalent to the prime ideal theorem for boolean algebras. *Bulletin AMS*, 60:387–388.

Herbelin, H. and Ilik, D. (2016). An analysis of the constructive content of henkin's proof of gödel's completeness theorem.

Herbelin, H. and Kirst, D. (2023). New observations on the constructive content of first-order completeness theorems. In *29th International Conference on Types for Proofs and Programs*.

Bibliography II

Kirst, D. (2022). *Mechanised Metamathematics: An Investigation of First-Order Logic and Set Theory in Constructive Type Theory*. PhD thesis, Saarland University. `https://www.ps.uni-saarland.de/~kirst/thesis/`.

Kirst, D., Hostert, J., Dudenhefner, A., Forster, Y., Hermes, M., Koch, M., Larchey-Wendling, D., Mück, N., Peters, B., Smolka, G., and Wehr, D. (2022). A Coq library for mechanised first-order logic. In *Coq Workshop*.

Kreisel, G. (1962). On weak completeness of intuitionistic predicate logic. *The Journal of Symbolic Logic*, 27(2):139–158.

Krivine, J.-L. (1996). Une preuve formelle et intuitionniste du théorème de complétude de la logique classique. *Bulletin of Symbolic Logic*, 2(4):405–421.

Krivtsov, V. N. (2015). Semantical completeness of first-order predicate logic and the weak fan theorem. *Studia Logica*, 103(3):623–638.

Bibliography III

Shillito, I. and Kirst, D. (2024). A mechanised and constructive reverse analysis of soundness and completeness of bi-intuitionistic logic. In *International Conference on Certified Programs and Proofs*, pages 218–229.

Simpson, S. (2009). *Subsystems of second order arithmetic*, volume 1. Cambridge University Press.

FOL Library: Proof Mode

```
    frewrite (ax_add_zero y).
    fapply ax_refl.
  - fintros "x" "IH" "y".
    frewrite (ax_add_rec (σ y) x).
    frewrite ("IH" y).
    frewrite (ax_add_rec y x). fapply ax_refl.
Qed.

Lemma add_comm :
  FAI ⊢ << ∀' x y, x ⊕ y == y ⊕ x.
Proof.
  fstart. fapply ((ax_induction (<< Free x, ∀' y, x ⊕ y == y ⊕ x))).
  - fintros.
    frewrite (ax_add_zero x).
    frewrite (add_zero_r x).
    fapply ax_refl.
  - fintros "x" "IH" "y".
    frewrite (add_succ_r y x).
    frewrite <- ("IH" y).
    frewrite (ax_add_rec y x).
    fapply ax_refl.
Qed.

Lemma pa_eq_dec :
  FAI ⊢ << ∀' x y, (x == y) ∨ ¬ (x == y).
Proof.
  fstart.
  fapply ((ax_induction (<< Free x, ∀' y, (x == y) ∨ ¬ (x == y)))).
```

```
1 goal
p : peirce
x, y : term
______________________________________(1/1)
 FAI
"IH" : ∀  x0, x`[↑] ⊕  x0 ==  x0 ⊕ x`[↑]
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
(σ x ⊕ y == y ⊕ σ x)
```

Messages ↗ Errors ↗ Jobs ↗

https://github.com/dominik-kirst/coq-library-undecidability/blob/fol-library/theories/FOL/Proofmode/DemoPA.v

FOL Library: Reification Tactic

```
Proof.
elim a using PA_induction.
- represent.
- eapply ieq_trans. 1:apply (add_zero_l (iS b)).
  apply ieq_congr_succ, ieq_sym, add_zero_l.
- intros d IH.
  eapply ieq_trans. 1:apply (add_succ_l d (iS b)).
  apply ieq_congr_succ. eapply ieq_trans.
  + apply IH.
  + apply ieq_sym, add_succ_l.
Qed.

Lemma add_comm a b : a i⊕ b i= b i⊕ a.
Proof.
elim a using PA_induction.
- represent.
- eapply ieq_trans.
  + apply (add_zero_l b).
  + apply ieq_sym, (add_zero_r b).
- intros a' IH.
  eapply ieq_trans. 2:eapply ieq_trans.
  + apply (add_succ_l a' b).
  + apply ieq_congr_succ, IH.
  + apply ieq_sym, add_succ_r.
Qed.
```

```
1 goal
D' : Type
I : interp D'
D_fulfills : forall (f : form) (rho : env D'),
             PAeq f -> rho ⊨ f
a, b : D'
______________________________________(1/1)
representableP 1 (fun a0 : D => a0 i⊕ b i= b i⊕ a0)
```

Messages Errors Jobs

https://github.com/dominik-kirst/coq-library-undecidability/blob/fol-library/theories/FOL/Reification/DemoPA.v

Abstract Formal Systems

Definition

A formal system $\mathcal{S} = (\mathbb{S}, \vdash, \neg)$ consists of:

- $\mathbb{S}$ is a set we consider the collection of formal sentences
- $\vdash$ is a semi-decidable subset we consider the provable sentences
- $\neg : \mathbb{S} \to \mathbb{S}$ is a computable function we consider negation, satisfying consistency:

$$\forall \varphi \in \mathbb{S}.\ \nvdash \varphi \vee \ \nvdash \neg\varphi$$

$\mathcal{S}$ is complete if for every φ either $\vdash \varphi$ or $\vdash \neg\varphi$.

Lemma (Refutation)

In a complete formal system we have $\nvdash \varphi$ *iff* $\vdash \neg\varphi$.

Proof.

That $\nvdash \varphi$ implies $\vdash \neg\varphi$ is by completeness, the other direction by consistency. □

Gödel à la Turing

Theorem

Let X be some undecidable set, for instance the halting problem K_Θ. If X reduces to the provable sentences $\vdash$ of a formal system $\mathcal{S} = (\mathbb{S}, \vdash, \neg)$, then $\mathcal{S}$ cannot be complete.

Proof.

1. Assume that X is undecidable and that $\mathcal{S}$ were complete.
2. Obtain that that $\nvdash \varphi$ iff $\vdash \neg\varphi$ by (Refutation).
3. Observe that the complement of $\vdash$ is semi-decidable.
4. Derive that $\vdash$ is decidable by (Post).
5. Conclude that X must be decidable by (Reduction).
6. Contradiction. □

Constructive Analysis of the Completeness Theorem

There are multiple dimensions at play:

- Syntax fragment (e.g., propositional, minimal, negative, full)
- Complexity of the context (e.g., finite, decidable, enumerable, arbitrary)
- Cardinality of the signature (e.g., countable, uncountable)
- Representation of the semantics (e.g., Boolean, decidable, propositional)

All contribute (not even independently) to the constructive status of completeness!